



#ШПАРГАЛОЧКИ

СОЗДАНИЕ САЙТОВ FRONT-END РАЗРАБОТКА

Дополнительный уровень

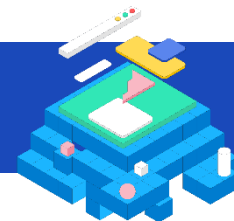
Материалы подготовлены отделом методической разработки

Больше полезных материалов и общения в нашем комьюнити
в Telegram: https://t.me/hw_school





Перенос данных в state.js



Перенос данных

В продолжение работы с **props** и **BLL** нужно вынести все данные, которые передаются через props, в отдельный файл.

Например, можно создать папку Data, а в ней - файл **state.js** и перенести в него все данные из **index.js**. Перед этим следует упаковать все массивы в один. И дополнительно распределить данные в зависимости от компонент, которые их используют. В результате получится объект, ключи которого обозначают компоненту, а значения - это снова объекты. Во вложенных объектах уже хранятся пары ключ - массив с данными.

Чем хорош этот подход - теперь данные хранятся более структурировано, а компоненты получают только те данные, которые им нужны.



Перенос данных

```
let state = {  
  profilePage: {  
    postsData: [...],  
  },  
  dialogsPage: {  
    dialogsData: [...],  
    messagesData: [...],  
  }  
}
```



Перенос данных

Теперь в index.js нужно импортировать state и передать компоненте App:

```
<App state={state} />
```

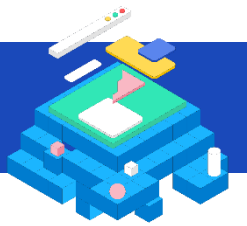
В компоненте App теперь из props нужно дополнительно извлекать state:

```
<Route path="/profile" render={() => <Profile postData={props.state.profilePage} />}>
```

```
<Route path="/dialogs" render={() => <Profile dialogsData={props.state.dialogsPage} />}>
```

Остается в компоненте Profile указать передачу Posts нужных данных:

```
<Posts postData={props.profilePage.postsData}>
```



Перенос данных

А компонента Dialogs распределяет данные на диалоги и сообщения уже внутри себя:

```
{props.dialogsPage.dialogsData.map((e) => <DialogItem  
name={e.name} id={e.id} />))}
```

```
{props.dialogsPage.messagesData.map((e) => <Message  
message={e.message} id={e.id} />))}
```



Структура

